REMARKS

Applicant respectfully requests reconsideration of this application, as amended, and consideration of the following remarks. Claims 1, 12 and 15 have been amended. Claims 2-4 and 13 have been previously canceled. Claims 1, 5-12 and 14-18 remain pending.

Claims 1, 5-12 and 14-18 stand rejected under 35 U.S.C. 102(b). Applicant traverses this rejection as set forth in more detail below. Applicant thanks the Examiner for withdrawing the prior rejections under 35 USC 101 and 112.

Amendments

Amendments to the Claims

Applicant has amended the claims to more particularly point out what Applicant regards as the invention. No new matter has been added as a result of these amendments.

Rejections

Rejections under 35 U.S.C. §102(b)

Claims 1, 5-12 and 14-18 stand rejected under 35 U.S.C. § 102(b) as being unpatentable over Hennessy et al (Computer Architecture: A quantitative Approach). Applicant respectfully traverses the rejection.

Hennessey teaches pipelining instructions and their hazards (e.g., stalls, etc.) that can occur in a pipeline.

Applicant believes that the Examiner has misunderstood Applicant's invention. Specifically, Applicant' invention allows implementing a swap of 2 instructions while maintaining a ***constant latency*** and the ability to launch swaps every clock cycle. Applicant's invention further uses only a single piece of hardware. (single set of pipeline devices) where the prior art requires two sets of pipeline devices to execute a swap.

Applicant's system and method provide that each swap is processed as a SAVE instruction and a RESTORE instruction in ***two consecutive clock cycles at all times***. In a first clock cycle, the Save instruction can move contents of an active

register to one of 8 or 4 window registers. In a next subsequent clock cycle, the Restore can move contents of a different window register to the active register. Save and Restore operations for the same swap instruction ***must*** happen in two ***consecutive*** clock cycles so as to not corrupt the data being saved or restored. Therefore every swap must take 2 clock cycles to complete.

Applicant's system and method provide that each swap is processed as a SAVE instruction and a RESTORE instruction in ***two consecutive clock cycles at all times*** and therefore providing a ***constant two clock cycle latency***. Applicant's invention includes circuitry and logic that achieves a flop function to complete the swap decoding without introducing additional clock cycles in the critical path.

Immediately subsequent swaps will also have a constant two clock cycle latency. Hence a constant latency of two clock cycles (for example: RESTORE for SWAP A, SAVE for SWAP B can happen simultaneously) is achieved by internal pipelining of swap instructions and throughput every cycle to launch a new swap instruction.

Hennessey's MOVI2S, MOVS2I will not have a constant latency as described and claimed herein. Hennessey's MOVI2S, MOVS2I and any immediately subsequent move instructions will have a ***progressive latency*** that progressively gets worse as described in Applicant's paragraphs 4-6 and Figure 1A-1B. as set forth herein:

> [4] Figure 1A shows a typical prior art processing window 100 used by a pipeline processor. The processing window 100 includes some number of register windows 110-117 (e.g., eight register windows reg0 – reg7). An active register window 120 is also included. The active register window 120 can be accessed by the processor. By way of example, as the pipeline progresses, the contents of register window 113 will be restored to the active register 120 so that the contents of register window 113 can be processed by the processor. However, typically the active register 120 is not empty and therefore the data in the active register 120 must be saved to the appropriate register window (e.g., register window 114) before the contents of register window 113 can be can be restored to the active register 120. Therefore, the contents in the active register 120 must be swapped (i.e., a save operation to register window 114 and a restore operation from register window

113).

[5] Figure 1B shows a timing diagram 150 of the pipeline 100. A clock 152 signal controls the timing of the various operations. When a swap request occurs, then in a first clock cycle the data from active window 120 is saved to the appropriate register window (e.g., register window 114) and then next cycle next cycle the content from the desired register window (register window 113) is restored to the active window 120.

[6] ***The swap requests typically operate in an acceptable manner as long as the swap requests do not occur too often***. However, multiple swap requests often occur immediately following one another. As a result, the pipeline can stall. By way of example, if a first swap request (i.e., swap A) is received, then a save A operation occurs, followed by a restore A operation, in the next clock cycle. A second swap request (swap B) is received immediately after the swap A request is received. However, two clock cycles are required to complete the save and restore operations required for the swap A request, therefore the swap B request must be delayed two clock cycles before the swap B request can be acted upon. ***As a result, the swap B request has a latency of 2 (two clock cycles between when swap B request was received and when restore B operation was completed). If a swap C request immediately followed the swap B request (i.e., during the third clock cycle), then the swap C request would have a latency of 4. Further, for n-subsequent swap requests, each swap request would have a n+1 latency factor.*** By way of example a tenth consecutive swap request (swap J) would have a latency of 11 (i.e., 11 clock cycles would pass between when the swap J request was received and when the restore J operation was completed). The latencies cause pipeline stalls where processing is delayed for that number of clock cycles. (Emphasis added)

Hennessey does not teach nor even suggest how to maintain a constant latency for Swap operations.

As to claims 1, 12 and 15, the Hennessy reference does not teach nor even suggest a pipeline architecture or a method for processing a plurality of swap requests comprising receiving a first swap request in a pipeline in a first clock cycle, and receiving an second swap request in a second clock cycle that immediately follows the

first clock cycle. The first swap request requests swapping active contents of a active register window with a first contents from a first register. Executing the first swap request includes executing a first save operation in the first clock cycle, wherein the active contents of the active register window is saved to a corresponding register and executing a first restore operation in a second clock cycle, wherein the second clock cycle immediately follows the first clock cycle, wherein the first contents of the first register are restored to the active register window. The second swap request requests swapping the first contents in the active register window with a second contents from a second register. Determining if the first register is a same register as the second register and executing the second swap request if the first swap request and the second swap request do not swap the same register. Executing the second swap request includes executing a second save operation in the second clock cycle. The first contents of the active register window is saved to first register at substantially simultaneously with the executing the first restore operation and executing a second restore operation in a third clock cycle, wherein the third clock cycle immediately follows the second clock cycle. The second contents of the second register are restored to the active register window, wherein the first swap request and the second swap request have a constant latency.

The Hennessey reference does not teach or even suggest each and every limitation as recited in claims 1, 12 and 15.

Accordingly, Applicant respectfully submits that Applicant's invention as claimed in claims 1, 12 and 15 and those claims that depend from one of claims 1, 12 and 15 are patentable over the Hennessey reference, and respectfully request the withdrawal of the rejection under 35 U.S.C. § 102(b).

SUMMARY

In view of the foregoing amendments and remarks, Applicant respectfully submits that the pending claims are in condition for allowance. Applicant respectfully requests reconsideration of the application and allowance of the pending claims.

If the Examiner determines the prompt allowance of these claims could be facilitated by a telephone conference, the Examiner is invited to contact George B. Leavell at (408) 749-6900, ext 6923.

Deposit Account Authorization

Authorization is hereby given to charge our Deposit Account No. 50-0805 (Ref # SUNMP351) for any charges that may be due or credit our account for any overpayment. Furthermore, if an extension is required, then Applicant hereby requests such extension.

Respectfully submitted,

MARTINE PENILLA & GENCARELLA, LLP

Dated: July 12, 2007

George B. Leavell
Attorney for Applicant
Registration No. 45,436
710 Lakeway Drive, Suite 200
Sunnyvale, CA 94085
(408) 749-6900 ext 6923